National Aeronautics and Space Administration

Where Should I Run My Job ?

Webinar Presentation, Dec. 7, 2011
NASA Advanced Supercomputing Division

Outline

1. Available Pleiades hardware (compute, front-ends, bridge nodes)
2. Running on pfe's and/or bridge nodes
3. Running on compute nodes
4. A sample PBS script
5. How to allow my job to run on any Pleiades processor type
6. Best practices with PBS jobs
7. PBS servers and queues

Supplemental Materials

- Job submission and monitoring
- Problems with submitting or running jobs

Preface: About NAS HECC Knowledge Base

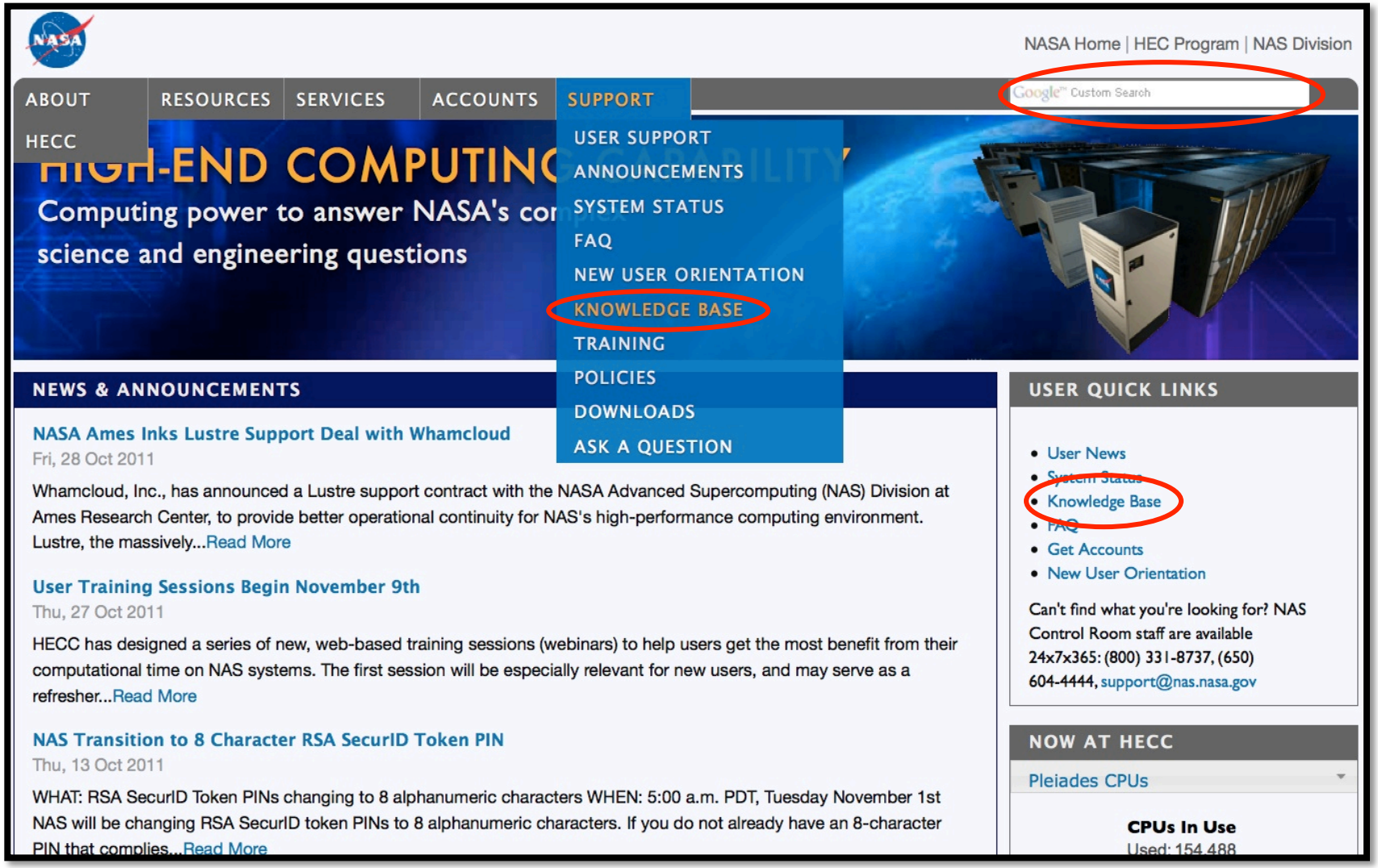
- Most information in this webinar is available in Knowledge Base (KB)

- URLs to get to KB
  - NAS HECC home page: http://www.nas.nasa.gov/hecc/ then click on Knowledge Base
  - KB home page: http://www.nas.nasa.gov/hecc/support/kb/

- Searching information in KB
  - Each of the ~200 web pages (or articles) has an ID number
  - Can search with text or ID

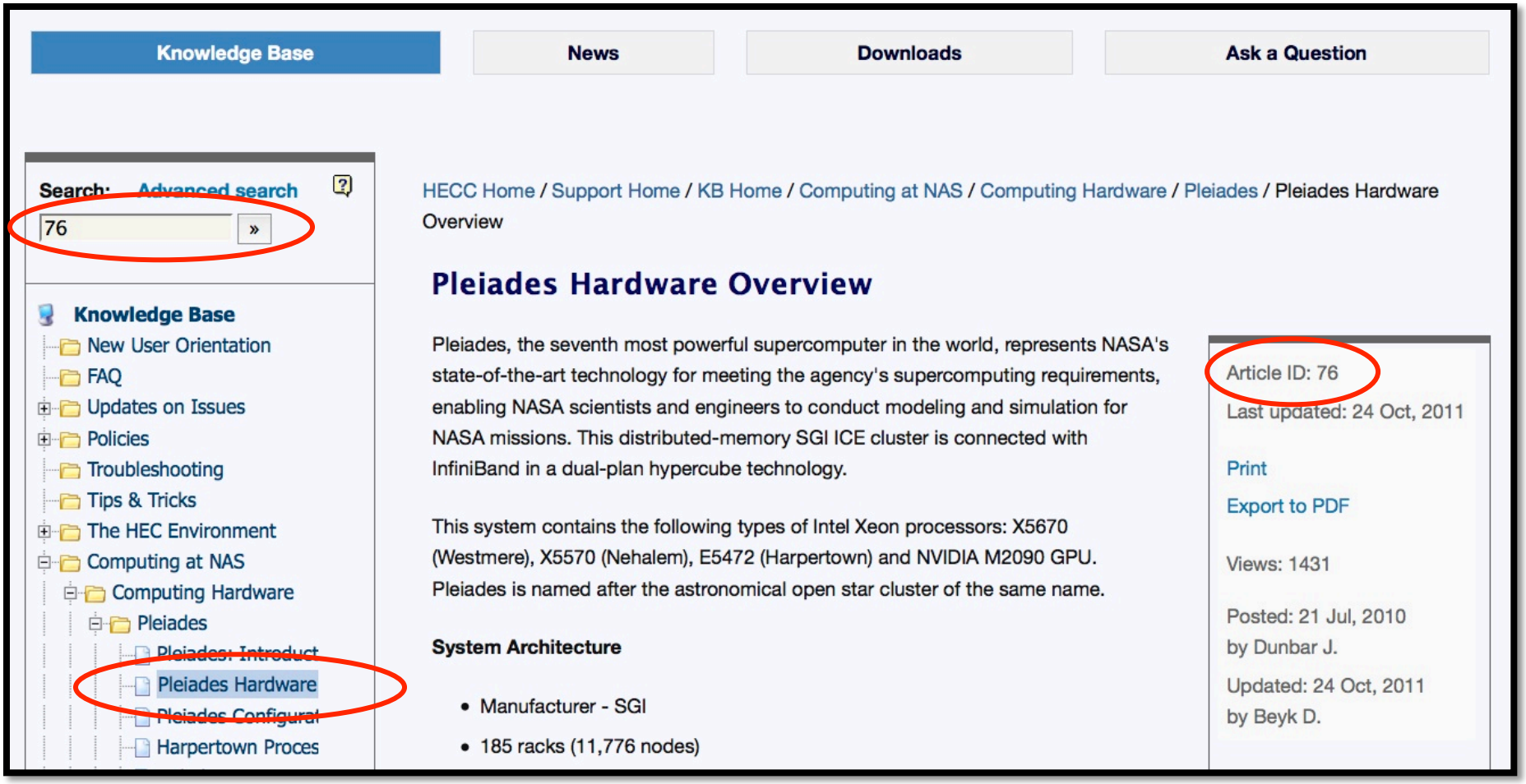
- Relevant articles will be mentioned in this webinar using their IDs

Searching info within HECC
http://www.nas.nasa.gov/hecc/



Searching info within KB

http://www.nas.nasa.gov/hecc/support/kb/



Available Pleiades hardware (compute, front-ends, bridge nodes)

➢ Compute nodes (KB 76 – 81)

- Hierarchy: ~185 racks -> ~185x64 nodes (r[1-222]i[0-3]n[0-15]) -> ~113,000 cores
- Three Intel Xeon processor types:

	Harpertown	Nehalem	Westmere
# of Racks	91	20	74
# of nodes	5,824	1,280	4,672
Cores/node	8	8	12
CPU speed	3.0 GHz	2.93 GHz	2.93/3.06 GHz
Memory/node	8 GB*	24 GB	24 GB
Memory/core	1 GB	3 GB	2 GB

1 rack has 64 nodes

* One Harpertown rack (rack 32) has 16 GB per node

- 64 Westmere nodes also include NVIDIA GPU (rack 219)
- Additional new hardware arriving Spring of 2012
- Compute nodes are for PBS jobs, not for logins

Available Pleiades hardware (compute, front-ends, bridge nodes) (cont.)

➢ Front-end pfe1-12 & bridge1-4 nodes (KB 76)

- Users can login to these nodes using ssh through sfe1-2 (see **KB 59** for details)
- They are shared with other users for interactive processing
- Differences among them

	pfe1-12	bridge1-2	bridge3-4
processor	Harpertown	Harpertown	Nehalem-EX
cores/node	8	8	32
CPU speed	3.0 GHz	3.0 GHz	2.27GHz
Memory/node	16 GB	64 GB	256 GB
Network	1 GigE	10 GigE	10 GigE
OS	SLES11SP1	SLES11SP1	SLES11SP1

Can run more processes on bridge[3,4] !!

bridge[1-4] have more memory and faster network than the pfe's !!

- After installing new hardware in Spring of 2012, some nodes may use different OS version – be careful with compiling codes and checking for available software modules. To check for OS version, use the command:
  uname –r or *cat /etc/sgi-release*

Running on pfe's and/or bridge nodes

➢ Usage Guidelines (KB 181)

– What should run
  - Pfe's: compiling, short debugging, testing, jobs submission
  - Bridge: same
    + pre/post-processing that need more memory
      Tecplot – no license restriction on number of users
      Do not set stacksize to unlimited
      IDL – 10 users (use "*module load idl/8.1*","*lmstat –a*" to check)
      Matlab – 8 licenses (use "module load matlab/2010b", "matstat" to check)
    + file transfers to/from Lou/Columbia

– What cannot/should not run
  - MPI jobs (mpiexec is jailed)
  - A job that needs (n cores x m minutes) > 160
  - A job that needs more memory than maximum limit

  limits: pfe's: 8GB; bridge[1,2]: 56 GB; bridge[3,4]: 192 GB

– Recommend to check if there is enough free memory on the node first
  - Use the command *top* or *cat /proc/meminfo* to check

Running on the pfe's and/or bridge nodes (cont.)

- Do not over-utilize these nodes. Jobs are monitored.
  - CPU time usage monitored by "topGun" script
  - Memory usage monitored by "query_wms" script

Sample emails to user when overutilization is detected

From: topGun@nas.nasa.gov
Subject: interactive job limit exceeded on pfe9...
Date: July 29, 2011 7:53:33 AM PDT
To: jsmith@nas.nasa.gov
the following interactive job(s) on pfe9 have been found, (but NOT killed).

USER	PID	PPID	TIME	COMMAND
jsmith	14496	1	**161:00**	a.out

Try fewer CPU's, less runtime -or- submit an interactive job session to PBS, via the command "qsub -I -l ncpus=1".

From: query_wms@nas.nasa.gov
Subject: [warning] jsmith@pfe1 over memory limit...
.. the following memory intensive job(s) found on pfe1
(but not killed)

USER	PID	RSS	TIME	COMMAND
jsmith	11690	**10.5G**	1:06	a.out

* policy: up to 8.0G of 15.6G memory may be used
* adjusted: 10.9G (136%) for jsmith's job 'a.out'
* formula: at 0.0% load, tracking jobs using 1+ cpu's

try using less memory, re-run on bridge[1-4] -or- submit an interactive PBS session, via "qsub -I -l ncpus=1:model=wes".

Running on the compute nodes

- What should run on the compute nodes
  - All production jobs (MPI, OpenMP, hybrid, serial)
  - Some development jobs or large/long debugging jobs
- Memory and processes/threads constraints

limit	Harpertown	Nehalem	Westmere
processes/node or threads/node	8	8 x 2*	12 x 2*
Memory/node	7.5 GB	22.5 GB	22.5 GB

*Due to hyper-threading; try with and w/o hyper-threading to see if there's benefit

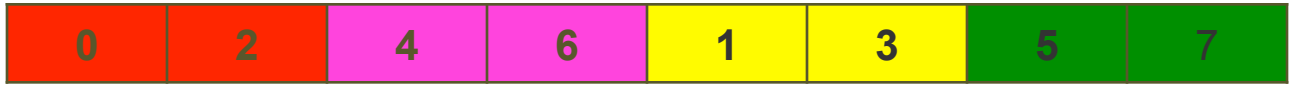
- When node out-of-memory detected by PBS, email from support@nas.nasa.gov is sent to user. Try reduce the number of processes per node or use processor type(s) with more memory. See **KB 216** for memory usage tips and memory monitoring tools
- The new hardware arriving in Spring will allow more memory/node (~30 GB/node)
- If you need large shared memory (~30 GB < mem < 4 TB) for production runs or have an OpenMP application that scales up to > 24 threads, apply for Columbia allocation

Running on the compute nodes (cont.)

– Pinning processes/threads on the processors (here 1 processor means 1 core)

Harpertown Configuration Diagram (**KB 78**)

Same color:
Share L2 cache



Nehalem Configuration Diagram (**KB 79**)

Same color:
Share L3 cache
and memory

Westmere Configuration Diagram (**KB 80**)

- CPUs that share identical resources (cache, memory, etc) are shown with the same color
- If processors are not fully occupied (e.g. running only 4 processes instead of 8 on Nehalem), different process placements will lead to performance differences. 4 processes assigned to 0,1,2,3 on neh and wes, will likely cause resource contention and bad performance
- With mbind.x –cs, processes are assigned to cores in a spread format to minimize contention. So, 0,2,4,6 will be used on neh; 0,3,6,9 used on wes. –v prints pinning information to output.
- mbind.x works for MPI (SGI MPT, Intel MPI, MVAPICH, OpenMPI), OpenMP and hybrid
- More info available at: /u/scicon/tools/bin/mbind.txt

Running on the compute nodes (cont.)

- Cost for running a PBS job on the three process types
  - Resources are allocated and charged by nodes, not by cores
  - Running jobs do not share the same nodes
  - Projects are given SBUs each year for running PBS jobs
  - Use *acct_query –pall –call –ujsmith –olow* to get SBUs charged to jsmith's jobs today

Hypothetical example: running with 128 processes

	# nodes	Walltime (hrs)	SBU rate	Cost (SBU Hrs)
Harpertown	16	20	0.45	144
Nehalem	16	11	0.8	140.8
Westmere	11	12	1.0	132

- Which process type allows a faster turn-around?
  - Tools for checking how many free nodes for each processor type:
    - *qstat –au foo [@pbspl1] (or @pbspl3)* [ ]: means optional
    - */u/scicon/tools/bin/qs [-s pbspl1] (or –s pbspl3)*
    - http://www.nas.nasa.gov/hecc/support/system_status.html (updated every 1-2 min)
  - Tools for checking how many nodes of each type queued jobs are waiting for:
    - *qstat –i –W o=+model,mission [@pbspl1] (or @pbspl3)*
    - */u/scicon/tools/bin/node_stats.sh*

A sample PBS script

➢ Sample PBS Script: (KB 175, 190)

resource list ->	#PBS –lselect=6:ncpus=8:mpiprocs=8:model=neh,walltime=24:00:00
comment out ->	##PBS –lwalltime=24:00:00
queue name ->	#PBS -q long
non-default GID ->	#PBS -W group_list=s0101
	#PBS -N my_job_name
merge stdout/err ->	#PBS -j oe
send email ->	#PBS -m e
no rerun ->	#PBS -r n
load modules ->	module load comp-intel/11.1.072 mpi-sgi/mpt.2.04.10789
job submitted from ->	cd $PBS_O_WORKDIR
implies –np 48 ->	mpiexec ./a.out > output
	# or
> recommended ->	# mpiexec -np 48 ./a.out > output

- Default GID can be found in /etc/passwd file (*grep username /etc/passwd*)
- PBS output/err file names: my_job_name.oxxxx (e.g., test.o12345, test.e12345)
  These files should be copied over by PBS to your $PBS_O_WORKDIR when job ends
- PBS automatically restarts a job (up to 3 times) which fails due to system issues
  or if PBS fails to detect job running out of memory

How to allow my job to run on all three processor types?

– Make sure that the executable can run on all three types

compile code without –xSSE4.2 (-xSSE4.2 runs on neh, wes only)
or
compile with -axSSE4.2,xSSE4.1 (**KB 99**)

– Prepare three separate PBS scripts
If not all cores are used, use mbind.x for pinning for good performance

For example, for a 12-process job, if each process needs 5 GB of memory
#PBS –lselect=12:ncpus=1:model=har (with 7.5 GB, can fit only 1 process)
#PBS –lselect=3:ncpus=4:model=neh (with 22.5 GB, can fit 4 processes)
#PBS –lselect=3:ncpus=4:model=wes

#mpiexec –np 12 ./a.out > output
mpiexec –np 12 /u/scicon/tools/bin/mbind.x –cs -v ./a.out > output

recommend adding /u/scicon/tools/bin in your path in .cshrc
set path = ($path /u/scicon/tools/bin)

Best practices for PBS jobs

- Redirect application output to a file (**KB 183**)
  - *mpiexec a.out > output* (can monitoring job progress, since output will be in $PBS_O_WORKDIR)
  - Without redirection, they are added to PBS stdout/stderr in *head-node%/PBS/spool*
  - /PBS/spool only has 1 GB of space. When it is filled up, PBS job may die. Won't get PBS stdout/err file.
  - New rule to be implemented soon: if PBS stdout/err > 100 MB, job is terminated
- Checkpoint long-running jobs
  - Checkpoint regularly to avoid losing data when job runs out of wall-time or fails due to system issues
  - If not sure how long it takes to finish the run, can checkpoint prior to running out of wall-time (**KB 199**, pbs_time_left.a)
  - Turn on #PBS –r n if you don't want PBS to restart your job for you
- Avoid combining large-cpu number crunching with small-cpu post-processing in 1 PBS job
  - If the small-cpu post-processing takes a long time, resource and allocation are wasted.
  - Submit a separate PBS job with fewer nodes for post-processing or do it on bridge nodes if possible
- Package multiple serial runs in a single PBS job (**KB 184**)
  - Submitting too many jobs slows down PBS job scheduling
  - Package serial jobs with *module load mpi-mvapich2/1.4.1/intel* and use technique described in KB 184

PBS servers and queues

➢ PBS (KB 126, 173-179, 186-190, 290)

- Two PBS servers:
  - pbspl1 is the default, it manages most queues
  - pbspl3 manages 2 queues (devel & gpu) (**KB 290**)
- Some queues (normal, long, debug, devel, gpu) are accessible to all Pleiades users
- Some are accessible only to certain missions/groups/users

  For example, armd_spl, kepler, vlong (16 days max)

  Manager(s) approval are needed to create/access these queues
- Reservation queues (**KB 176**) require HECC manager approval; set up by sysadm

  Special circumstances which warrant the use of reservation queues:
  - time critical simulations over a period of time
  - very large simulations (e.g., 20,000-core jobs)

  Charged even when not used

PBS servers and queues (cont.)

- **PBS queues accessible to all Pleiades users: (KB 187, 290)**

```
pfe1% qstat -Q  (or qstat -Q @pbspl1)

Queue     Ncpus/       Time/
name        max/def     max/def      pr
========  =====/===  ======/======   ==
debug      1025/   8   02:00/ 00:30   15
long       8192/   8  120:00/ 01:00    0
normal       --/   8   08:00/ 01:00    0
```

```
pfe1% qstat -Q @pbspl3

Queue     Ncpus/      Time/
name        max/def    max/def     pr
========  =====/===  =====/=====   ==
devel     4800/   1  02:00/   --    0
gpu         --/   8  08:00/01:00    0
```

- Debug queue has higher priority
- If queue name is not specified during job submission:
  - walltime <= 8 hours, default to normal queue
  - walltime > 8 hours, but <=120 hours, long queue
  - walltime > 120 hours, job will be rejected
- har, neh, wes available for all these queues
- if processor type not specified, default to harpertown

- 8 Westmere racks (512 nodes) set aside 7x24 for devel queue
- Served by pbspl3, devel queue allows faster job scheduling and turnaround
- 1 running job per user on devel queue
- no production work on devel queue
- Use gpu queue (64 nodes) if you have a CUDA application for running on GPU

- Use *qstat –fQ queue_name@server_name* to get more info about each queue

Summary

- **Where should I run my jobs?**
  - Front-end and bridge nodes resources and usage guidelines
  - Compute nodes resources and factors to consider for PBS jobs
    - three processor types
    - cores consideration
    - memory consideration
    - cost consideration

Harpertown	Nehalem	Westmere
8	8x2*	12x2*
7.5 GB	22.5 GB	22.5 GB
SBU 0.45	0.8	1.0

(*acct_query*)

    - availability consideration (*qstat –au foo, qs, qstat –i, node_stats.sh*)
    - pinning for better performance (*mbind.x –cs –v*)
  - Sample PBS scripts and Best Practices for PBS jobs (redirect output !!)
  - PBS queues to submit your jobs to (*qstat –Q*, use devel for development)

- **For more information, see KB articles listed in this webinar**

- **For assistance**: call 1-800-331-8737 or 650-604-4444
  or email support@nas.nasa.gov

Slides Prepared by Sherry Chang

pdf and recording of this webinar will be available shortly at:
http://www.nas.nasa.gov/hecc/support/training.html

Next webinar
"I/O: Tips and Techniques"
tentatively scheduled on Jan. 11, 2012

Suggestions for future webinar topics are welcomed

Job Submission and Monitoring

➢ Submitting your PBS job

– Batch jobs

% *qsub job_script* (for pbspl1 only) or %*qsub –q devel@pbspl3 job_script*

12345.pbspl1.nas.nasa.gov 2468.pbspl3.nas.nasa.gov

– Options in the command line override those in job_script

% *qsub -lselect=4:ncpus=12:model=wes job_script*

12347.pbspl1.nas.nasa.gov

– Interactive PBS job (good for debugging or development work)

% *qsub -I -lselect=4:ncpus=12:model=wes* (Note: no job script should be included here)

➢ Monitoring your PBS job

% *qstat –u jsmith*	(show all running/queued jobs by jsmith - pbspl1)
% *qstat @pbspl1 @pbspl3 –W combine –u jsmith*	(show all jsmith's jobs served by pbspl1 and pbspl3)
% *qstat –nu jsmith*	(show nodes for running jobs - pbspl1)
% *qstat –f 2468.pbspl3*	(show details of a running/queued job - pbspl3)
% *qstat –s 12347*	(gives one line status explanation of a job - pbspl1)
% *qstat –i*	(show all queued jobs sorted by priority – pbspl1)
% *qstat –xu jsmith*	(show all finished/running/queued jobs of jsmith)
% *qstat –xf 12345*	(show details of a finished/running/queued job)

Problems with submitting or running jobs

- **Can't Submit: (KB 197)**
  - *qsub: Job exceeds queue and/or server resource limits*
    (for example, asking for 3 hours in devel or debug queue)
  - *qsub: Job rejected by all possible destinations*
    (for example, qsub –lwalltime=140:00:00)
  - *qsub: Unauthorized Request*
    (AUID/GID not in ACL, no more allocation)
  - *qsub: Bad GID for job execution*
    (AUID not in the GID used for submission)
  - *qsub: would exceed complex's per-user limit*
    (exceed 300 total jobs in Q or R state)

- **Job waiting in queue, not running (KB 198)**
  - get a one line status information (use *qstat –s jobid*)
  - job is waiting for resources (use *qstat –i* to check job position)
  - job will never run due to impossible resource request (eq. submit to devel asking for har)
  - job would exceed mission CPU share (use *%qstat –W shares=-* to check, **KB 168**)
  - system going into dedicated time (check email and MOTD announcement)
  - your home filesystem or /nobackup is down (possible system problems)
  - over quota on home or /nobackup filesystem (hard limit)
  - job on Hold (for example, held by sysadm or after being run 3 times but failed)

Problems with running jobs (cont.)

- **What to do if my job ran, but failed or stalled ?**

– check PBS stdout/stderr (in $PBS_O_WORKDIR) for possible error messages

command not found (check if module loaded? path included? hidden character?)
Disk quota exceeded (delete files to go under disk/inode quota hard limit)

SIGFPE(8) – Floating-point exception
SIGSEGV(11) – Segmentation violation
(recompile code with debugging options –O0 –g –check –traceback)

– check email – if PBS detects job encountering OOM
(monitor job to find out memory needed; increase memory request) (**KB 216**)

– possible system problems (such as IB network, Lustre file-system, bad cpus or memory)
MPI: rank 100: r1i0n3 IB board mlx4_0 port 1 had fault
(report problems; resubmit to different processor type)
(the systems team adds check in the PBS prologue to offline low-memory nodes)

– ask for help (send email to support@nas.nasa.gov)